International Journal of Computer Science Engineering
and Information Technology Research (IJCSEITR)
ISSN(P): 2249-6831; ISSN(E): 2249-7943
Vol. 4, Issue 3, Jun 2014, 171-178
© TJPRC Pvt. Ltd.

GOOGLE GLASS AND GLASSWARE DEVELOPMENT USING REST ARCHITECTURE

ROHIT SAXENA[1], VIPUL KUMAR[2], ASHISH GUWALANI[3], ANIL DUBEY[4] & RAKESH RATHI[5]

[1,3,4]Faculty, Department of CS & IT, Govt. Engineering College Ajmer, Rajasthan, India

[5]Assitant Professor, Department of CS & IT, Govt. Engineering College Ajmer, Rajasthan, India

[2]Student, Btech IV Year, Department of CS & IT, Govt. Engineering College Ajmer, Rajasthan, India

ABSTRACT

This paper demonstrates the establishment of Google Glass and invites a discussion over the proposals of eliminating its potential drawbacks. The paper proceeds to the architecture and conceivable mechanism of Mirror and REST APIs after an interpolation to the arrival of Google Glass and its celebrated improvements over upgrades. Besides, the paper deliberates the idea of Glassware development with Mirror API which is layered onto the architecture of RESTful programming. The paper advocates the employment of REST structure for Mirror API but also argues for amelioration of the API architecture with progressive intent. The paper contemplates the RESTful behavior of Mirror API and helps the audience to perceive the underlying mechanism of Google Glass and Glassware. As much as the depiction of existing architecture of conferred APIs is entertained, there is a crowning spot in the paper for the betterment of existing paradigms.

KEYWORDS: Google Glass, Glassware, Mirror API, REST Architecture, Glass Architecture, Google Glass, Glassware, Mirror API, REST Architecture, Glass Architecture

INTRODUCTION

Google Glass

The dream of thousands of years of humanity is about to come true. The dream being dreamt from Leonardo Da Vinci [1] to Sergey Brin [2]. Now, we have a device that can work as they say (though, not as they think). Developed by Google X labs and manufactured by Faxconn, this 50g weighted gadget is a transparent Heads-Up Display (HUD), a prototype for Augmented Reality (AR), worn over one eye. The idea of sending a message or searching for "the length of Brooklyn Bridge" or translating a phrase to a foreign language just by saying it is exhilarating in itself [3]. The HCI community is more focalized on broadening the field of user centred design and user experience in recent times. A device worn over head, doing whatever the user says, intensifies the thought of a technology driven future. The first appearance of Glass was made in a public event "Foundation Fighting Blindness" in San Francisco on April 5, 2012 by Sergey Brin [4]. The emergence of Glass is absolutely not a sudden thought for humanity. The standards for a device like this are being materialized for long by fiction. The efforts are being made for the patents of Augmented Reality for a long time. In such circumstances, Google has done enough to intimidate its market competitors in Augmented Reality for quite a time. The Land Warrior system by U.S. army is a considerable example for the implemented projects in Augmented Reality but the HUDs hadn't made an appearance on this significant level until now. The emergence of Glass is also being considered the result of Sergey Brin's obsession with fiction as the Glass is advertised as the brain child of Sergey Brin [5].

GLASSWARE

Glassware is a name given by Google to the web services which can send content to Glass or receive content from Glass. So while using practically, almost every app of Google Glass will be a Glassware. E.g. Gmail lets the user know about new message arrival by notification. [6]

Developing Glassware

Google has always been helpful when it comes to the development of apps for its operating system (Android). Google released a sneak-peak of Glass Developer Kit (GDK) and a Mirror API before even releasing a stable consumer version of Glass [7]. The motive of releasing such helpful tools is to motivate more and more programmers to join the agenda of make Glass a really helpful assistant. Smartphones (Especially Google's Android) has made it very common and convenient for programmers to develop apps for different mobile operating systems. At present, the success criteria of a device like this is directly affected by the quality apps or addons available for it in market. So Google (just like other big operating system owners) is encouraging developers to develop quality apps for its operating system.

Glass Development Kit

GDK will be the most helpful way of developing apps for those who are already immersed in android development. It is supposed to be an Android based development kit for standalone Glass apps. In the terms of definition of Glassware, stand-alone Glass apps are not included in certain group of Glassware. So the only mechanism of developing "neat" Glassware is through using Mirror API [8].

The Mirror API

It is a RESTful API designed to work with languages supporting HTTP for developing Glassware that are independent of Glass operating system. The development model of Mirror API is very different from that of Android and even traditional web apps. So it might be the "new" development API for Glassware development. The principal component in REST architecture, and so in Mirror API, is the server. In an API that complies with the paradigms of REST, client does not need to know anything about the architecture. Rather, the server handles the requests and provides services accordingly. The Glassware developed through Mirror API aren't actually native OS apps and they don't run directly on the Glass. Rather, they run on a server. The interaction to Glass is made by inserting interactive cards to the Glass Timeline. So every user interaction consists a round-trip from Glass to server and back. The API is used to present a web service at cloud server and can be controlled by querying the server to send or retrieve data. The future of Glassware resides in the implementation and improvement of Mirror API. It is quite predictable that most developers familiar with Android development and not familiar with REST architecture will prefer the Glass Development Kit (GDK) over the Mirror API but still there is a large number of interested programmers fluent in other languages. REST accommodates all the languages those can work with HTTP. So it covers quite a range of developers around the world.

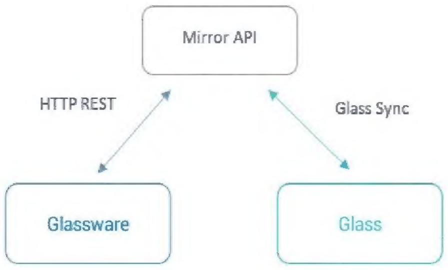

Figure 1: A Simple Representation of Interaction of Glassware to Glass through Mirror API

REST is used as a Web service protocol for Google Glass. In simplest terms, REST is merely a set of rules and paradigms which explain the use of HTTP. HTTP is primarily oriented around verbs and resources. The verbs are applied to resources in order to get access and process the instructions provided by the server. The basic HTTP verbs, however, aren't enough to handle such a large scale architecture. Besides, the simpler verbs (Especially GET) aren't capable of managing high data load.

The intermediary element while applying queries and getting results from a Glassware is the server. The API at server can transform the message descriptions and semantics according to the preferences of user and developer, and the Timeline cards referring to particular Glassware are made visible to user. The REST API advocates to actualize this behaviour [9]. The architecture of Google Glass, based on Mirror API, is shown in Figure 3. The user has to authenticate itself by submitting the credentials which are authorized by Google OAuth 2.0 servers. In this process a code is generated at Google OAuth 2.0 servers and it is exchanged with an access_token. Once authorized, Glassware can communicate with Mirror API over HTTPS by sending messages. JSON format is used by Mirror API in this process [10].

But the illustrious prospect here is to note that it requires a constantly reliable internet connection through the process. It's true for any service based on REST architecture that it needs to be online through the process. It augments the field of problems in the cases where Internet connectivity might not be very consistent.

Google cloud server is synchronised with Google Glass. The Cloud server can take requests and serve to both Glass device and Glassware through Mirror API. As soon as the server gets a request for user involvement regarding the Glassware cards it has, it inserts them on the Glass Timeline which then displays the Cards to the user accordingly.

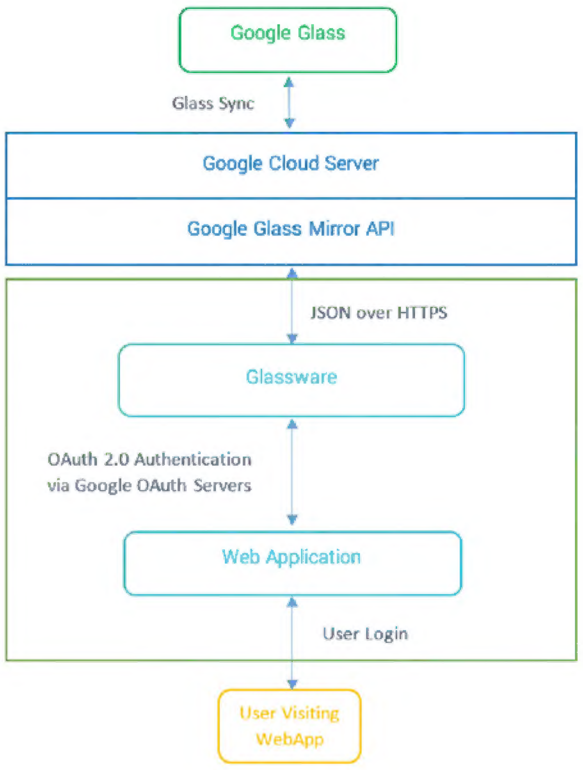

Figure 2: Architecture of Google Glass with Mirror API

DISCUSSIONS AND PROPOSALS

For Google Glass

Problem: The most critical prospective being feared with the emergence of Google Glass is the Security and privacy concern. “The possession of great power necessarily implies great responsibility”. In a world where knowing is owning, we all must fear someone who can know about anything instantly without even letting other people know about it. As much as the information on hand is smart and pragmatic, it can always be used to accomplish negative intents. Google has already taken steps to ban facial-recognition apps but it’s not adequate to take out the fear from its audience’s minds. Another privacy comprehension is concerned with the ease of capturing photos and recording videos brought in by Glass. People are afraid by the fact that the person simply looking at them might be capturing a photo or recording a video of their activities. However, Google remarks that it won’t be a problem as the user will have to command the Glass either by voice or touchpad. But it doesn’t make it any less intimidating. The involvement of the fact that it’s too easy for users to invade in the privacy of others’ lives is still alarming. This might be proved a major downside for Glass’ acceptance and reputation. If the trend is set to looking suspiciously at the Glass users, the Glass will have to pay a grand price for its reputation.

Proposal: The hardware capabilities should be limited to not let a security concerned app, like a facial recognition app, run on Glass. The measures should be taken to not let developers even develop such type of applications using development kits.

Problem: Another concern, when it comes to an “addictive” digital device which will assist the user for probably more time than any other digital device, is health involvement. The EMF radiation being radiated by such mobile devices has been an issue of interest for a long time. The issue has been controversial and according to some articles the device (mobile) radiations don't have enough energy to directly affect the DNA but the World Health Organisation nevertheless classifies the Radio-Frequency fields as "possibly carcinogenic" [11].

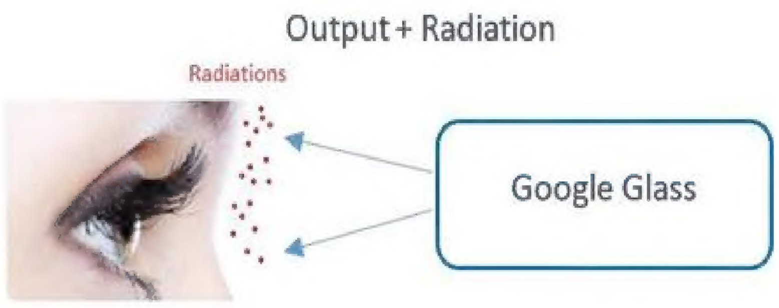

Figure 3: Glass Emitting Radiations

Proposal: The device has lowered the radiations but it will be further better to provide additional mechanisms for the health assurance. The device will be much closer to brain than any other digital device. Maybe in future, a protection would be better add-on to Glass which can absorb the radiations and make them even less harmful. Some interface can be introduced to minimize the effect of radiations. The dream of a technology driven world certainly raises potential concerns to the people about over-use of the technology.

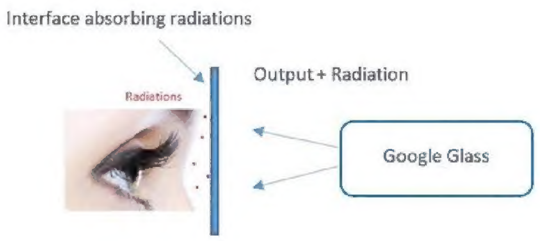

Figure 4: Interface Absorbing Radiations Emitted From Glass

Problem: The over-use of technology also concerns people about losing their social attachments. The Glass certainly makes it easier to connect and communicate with other Glass users but what about people standing in gatherings and talking to their "Glasses". It's way too non-social in the traditional world.

Proposal: The remedy is with users to limit the use of technology to suitable fields and don't let it invade into certain aspects of life.

For Mirror API and REST Problem

While RESTful programming is highly beneficial, there are still some aspects to be taken care of. The most of mistakes made while implementing REST, are the developers' choices of implementation. The most common misuse of REST by developers is to expose the chunks of inner data structures or databases. As stated earlier, REST substantially favours the idea of abstraction when it comes to maintaining data and objects.

Proposal: The idea is to expose only higher level objects that are relevant to the system and not to bother about those on lower level. The exception and error handling is the straight way to go for lower level interactions when met with any kind of inconsistency. With the introduction of REST services in Glass apps, a large number of new programmers are expected to join the REST community. So the aspect of abstraction in the Mirror API is to be made more comprehensible to the developers.

Problem: The RESTful services require a reliable Internet connection for a real-time experience. On a device like Google Glass which is always mobile, the Internet connectivity can be a problem in certain areas. And without a sustaining network connection, the REST structure, thus the Mirror API, thus the Glassware cannot perform to the anticipated value.

Proposal: The approach can be taken to introduce an offline behaviour for emergency units like medical. The data can be transferred when Glass is online.

Problem: Another drawback of REST is that it doesn't have any common standard described for formal REST descriptions. REST is not a framework, it's an architectural style. So it is descriptive in any language supporting HTTP. The support for versatile languages is an advantage for its adaptability but when it comes to depicting a sophisticated standard of interpretation, REST is too confined for it.

Proposal: It needs to be versatile but it also needs a standard set on a whole level to make programmers more directed towards an approach. So a descriptive standard should be adopted for RESTful architectures to be described.

Problem: Besides, HTTP verbs are too constrained to be used in an architectural style dealing with such a large amount of data. REST is not cultivated to handle large amount of data. Being developed only on the support of HTTP is the origin of all the look-downs with REST.

Proposal: If an improvement is to be made to REST, the suggested idea will be to include the paradigms of another protocols. By this, the architecture will be enriched with additional requests and it might be able to serve concurrent requests and process large data.

CONCLUSIONS

As much as it is important to understand the underlying architecture of Google Glass and Mirror API, it is also vital to apprehend that their drawbacks are confrontational and significant enough to change the course of their development and then their acceptance among the audience. REST is probably the most talked and celebrated architecture in itself but it also has to go under some enlargements to be sufficient alone for large scale development. The proposed solutions are mostly hypothetical and carry a preliminary indication about what can be done. The subject is an ongoing research and expects better aspects to be established in future.

ACKNOWLEDGEMENTS

The authors would like to thank everyone involved in this paper's content in any manner. We would like to thank those who believed in us and always motivated us. Also, we owe a big thanks to those who motivated us to prove them wrong.

REFERENCES

1. Leonardo Da Vinci. The sketches by Da Vinci depicting a device very similar to Google Glass. http://www.mashable.com/2013/03/31/da-vinci- invent-google-glass/
2. Sergey Brin on Ted's show, talking about the origin of Google Glass. www.ted.com/speakers/sergey_brin

3. “What it does”, Quick start with Glass. http://www.google.com/glass/start/what-it-does/

4. "Augmented reality glasses debut on Google co-founder's face". Los Angeles Times. http://articles.latimes.com/2012/apr/06/nation/la-na-nn- sergey-brin-project-glass-20120406

5. Isabel Pedersen, Douglas Trueman, “Sergey Brin is Batman”: Google’s Project Glass & the instigation of computer adoption in popular culture

6. Google Glass developers https://developers.google.com/glass/

7. “The GDK APIs aren’t finalized”. Official developer page. https://developers.google.com/glass/

8. About Mirror API. Official Google documentation. https://developers.google.com/glass/develop/mirror/index

9. Roy Thomas Fielding, “Architectural styles and the design of network- based software architectures” dissertation. P. 82. 5.1.6 Layered system.

10. Google I_O 2013 - Building Glass Services with the Google Mirror API. Video www.youtube.com/watch?v=CxB1DuwGRq

11. “World Health Organization, “IARC classifies radiofrequency electromagnetic fields as possibly carcinogenic to humans”, Press Release N° 208, 31 May 2011.